Supported by the
National Science Foundation
Engineering Research Center
Program (NSFD CD 8803012),
Industry and the University

Extension of the Fixed-Rate Structured Vector Quantizer to Vector Sources

by R. Laroia and N. Farvardin

TR 91-106

Extension of the Fixed-Rate Structured Vector Quantizer to Vector Sources [†]

by

Rajiv Laroia and Nariman Farvardin
Electrical Engineering Department
and Systems Research Center
University of Maryland
College Park, Maryland 20742

Abstract

The fixed-rate structured vector quantizer (SVQ) derived from a variable-length scalar quantizer was originally proposed for quantizing stationary memoryless sources. In this paper, the SVQ has been extended to a specific type of vector sources in which each component is a stationary memoryless scalar subsource independent of the other components. Algorithms for the design and implementation of the original SVQ are modified to apply to this case. The resulting SVQ, referred to as the extended SVQ (ESVQ), is then used to quantize stationary sources with memory (with known autocorrelation function). This is done by first using a linear orthonormal block transformation, such as the Karhunen-Loève transform, to decorrelate a block of source samples. The transform output vectors, which can be approximated as the output of an independent-component vector source, are then quantized using the ESVQ. Numerical results are presented for the quantization of first-order Gauss-Markov sources using this scheme. It is shown that the ESVQ-based scheme performs very close to the entropy-coded transform quantization while maintaining a fixed-rate output and outperforms the fixed-rate scheme which uses scalar Lloyd-Max quantization of the transform coefficients. Finally, it is shown that this scheme also performs better than implementable vector quantizers, specially at high rates.

Index Terms: Fixed-rate coding, structured vector quantization, independent-component vector sources, correlated Gaussian sources.

[†] This work was supported in part by National Science Foundation grants NSFD MIP-86-57311 and NSFD CDR-85-00108, and in part by a grant from General Electric.

I. Introduction

A fixed-rate structured vector quantizer (SVQ) for quantizing stationary memoryless scalar sources was introduced by Laroia and Farvardin in [1]. The SVQ is derived from a variable-length scalar quantizer and bridges the gap between the performance of the optimal entropy-constrained scalar quantizer [2]-[4] and the error-minimizing fixed-rate Lloyd-Max quantizer (LMQ) [5],[6] while maintaining a fixed-rate output. The design and implementation complexities of the SVQ are only polynomial in block-length and are manageable even for relatively fine quantization (high bit-rates) at large block-lengths. For memoryless sources, the SVQ is an attractive alternative to LBG vector quantizers (LBGVQ) [7]. This is especially true at high rates where, for comparable performance, the LBGVQ complexity (which grows exponentially with the rate-dimension product), is significantly higher than the SVQ complexity.

In this paper, the SVQ has been extended to the quantization of a specific type of vector sources and the resulting quantizer is referred to as the extended SVQ (ESVQ). Each component of the vector source considered here is a stationary memoryless scalar subsource independent of the other components; the scalar subsources can have distinct probability distributions. Such sources are encountered in the transform coding of images, speech and other signals. An algorithm for the suboptimal design of the ESVQ is developed along with fast algorithms for codebook search and codevector encoding/decoding.

The ESVQ for independent-component vector sources is then used to quantize stationary sources with memory. This is done in the context of block transform quantization. The idea of block transform quantization was introduced by Huang and Schultheiss in [8] for quantizing stationary correlated Gaussian sources. They use the Karhunen-Loève transform, which is a decorrelating orthonormal linear transformation, to decorrelate a block of source samples. Each transform output vector is quantized using a bank of fixed-rate Lloyd-Max scalar quantizers. Farvardin and Lin in [9] have improved upon these results (in the rate-distortion sense) by using a bank of variable-length encoded scalar quantizers in place of LMQs. The price of this improvement is a variable-rate output which makes the system difficult to implement because of the buffer overflow/underflow and error propagation problems associated with variable-rate systems. Since the output vectors of the decorrelating transform can be approximately modeled as the output of an independent-component vector source of the kind described earlier, they can be quantized using the

ESVQ. This results in an improvement over the HS scheme while maintaining a fixed-rate output.

Since an ESVQ-based system enjoys the same structural properties as the SVQ, its computational complexity is significantly lower than that of an LBGVQ for the same codebook size. Therefore, with the ESVQ-based system dramatically larger codebooks can be used before the complexity or memory requirements render it impractical. This means that for a given encoding rate, the block-length of the ESVQ-based system can be much larger than that of the LBGVQ and hence smaller distortions may be achieved (especially if the source is heavily correlated). Specific examples given for stationary Gauss-Markov sources will confirm this claim.

The rest of this paper is organized as follows. A brief description of the SVQ for memoryless sources is provided in Section II. The extension of the SVQ to independent-component vector sources (ESVQ) is considered in Section III. An algorithm for a sub-optimal design of the ESVQ is presented in Section IV while Section V addresses the implementation issues which include codebook search and codevector encoding. Quantization of stationary sources with memory is considered in Section VI along with numerical results for quantizing first-order Gauss-Markov sources and appropriate comparisons with several other schemes. In Section VII, the ESVQ is compared with an SVQ-based system for quantizing independent-component vector sources. Finally, a summary and conclusions are presented in Section VIII.

II. The Structured Vector Quantizer

We present here a brief description of the SVQ along with a summary of some of the important results. The basic idea behind the SVQ is simple and can be qualitatively described as follows. Consider an entropy-constrained scalar quantizer (ECSQ) the output of which is encoded by a variable-length code in which the length of each codeword is given by the negative logarithm of the probability of the corresponding quantization level. For simplicity assume that this results in integer lengths. If such a variable-length encoded ECSQ (VLE-ECSQ) is used to quantize a block of samples from a memoryless source, the total length of the output reflects its probability — smaller length means higher probability. When the block-length is large, with a high probability, the total length of the output is close to the *typical length* ($\equiv$*block-length* $\times$ *ECSQ entropy*). This suggests that if, for a large block-length, the output length of a VLE-ECSQ is forced to lie in a small neighborhood of the typical length (possibly introducing additional distortion), then with a high probability the quantizer outputs can be encoded without any additional distortion. Thus, if all the possible VLE-ECSQ outputs that satisfy the total length constraint are counted and encoded with fixed-length codewords, the system will result in an average distortion close to that of the VLE-ECSQ. Although for large block-lengths, the performance of this system approaches that of the VLE-ECSQ [1], for any fixed block-length it is preferable to encode all the VLE-ECSQ outputs of total length equal to or less than the typical length. This is because smaller length outputs are more probable. This idea provides the motivation for the following structure of the SVQ.

Let $\mathcal{S}$ denote an n-level scalar quantizer with the set of quantization levels $\mathcal{Q} \equiv \{q_1, q_2, \dots, q_n\}$ and the corresponding set of lengths $\mathcal{L} \equiv \{\ell_1, \ell_2, \dots, \ell_n\}$ where ℓ_i are positive integers and could, for example, be (but are not restricted to) lengths of the codewords if a variable-length code such as the Huffman code is used at the output of $\mathcal{S}$. If each component of an m-vector is separately quantized using $\mathcal{S}$, the quantized vector is in $\mathcal{Q}^m$, which is a rectangular grid of n^m points. Out of these n^m points only those that have a total length (sum of the component lengths) no greater than a threshold L are chosen as codevectors of the SVQ $\mathcal{V}$. The threshold L is chosen such that the codebook contains no more than 2^{mr} codevectors where r is the desired rate of $\mathcal{V}$ in bits per sample. A formal definition of the SVQ now follows [1].

Definition 1: An m-dimensional SVQ $\mathcal{V}$ derived from the scalar quantizer $\mathcal{S} \equiv (\mathcal{Q}, \mathcal{L})$ is a vector quantizer with a codebook $\mathcal{Z}$ given as,

$$\mathcal{Z} = \{\mathbf{z} = (z_1, z_2, \dots, z_m) \in \mathcal{Q}^m : \ \ell_{f(z_1)} + \ell_{f(z_2)} + \cdots + \ell_{f(z_m)} \leq L\}, \tag{1}$$

where the index function $f : \mathcal{Q} \to J_n \equiv \{1, 2, \dots, n\}$ is defined by, $f(q_i) = i, \ i \in J_n$.

From this definition it is apparent that any permutation of a codevector of the SVQ is also a codevector (all permutations of a codevector have the same total length); the SVQ can hence be considered as a modification of the permutation coder [4] where more than one *type*[1] of codevectors are present in the codebook. The pyramid vector quantizer introduced by Fischer in [10] for Laplacian sources is, in a loose sense, a special case of

[1] Here the type of a codevector refers to the set of relative frequencies of occurrence of the quantization levels of $\mathcal{S}$ in the codevector.

the SVQ when the quantization levels are uniformly spaced and $\ell_i = |q_i|$ for all $i \in J_n$. Similarly, Fischer's spherical vector quantizer [10],[11] for Gaussian sources is a special case of the SVQ when quantization levels are uniformly spaced and $\ell_i = q_i^2$. Fischer has shown that in the limit of large block-length, the pyramid (spherical) vector quantizer performance for a stationary memoryless Laplacian (Gaussian) source approaches that of the optimal ECSQ at the same entropy (rate). For other sources, the ECSQ performs better than both pyramid and spherical vector quantizers. Under certain assumptions, it is shown in [1],[12] that the performance (signal-to-noise ratio) of the SVQ is also bounded above by the performance of the optimal ECSQ at the same entropy, but the SVQ asymptotically achieves the performance of the ECSQ as the block-length increases. Algorithms for the design and efficient implementation of the SVQ are also presented in [1],[12]. Results on the performance of the SVQ show that the SVQ effectively bridges the gap between the performances of the optimal fixed- and variable-rate scalar quantizers [1]. At high rates the SVQ also performs better than implementable vector quantizers, especially for sources with a sharp-peaked, broad-tailed density. The SVQ is robust in the presence of channel noise and outperforms both fixed and variable-rate scalar quantizers for a wide range of channel bit-error probabilities [1].

III. Extension to Independent-Component Vector Sources (ESVQ)

In this section we extend the SVQ for memoryless sources to a specific type of vector sources. The vector sources considered here are assumed to have independent components and each component is assumed to be a stationary memoryless scalar subsource; each scalar subsource can have a distinct probability distribution. In the special case when all subsources are identically distributed, the ESVQ reduces to the SVQ for memoryless scalar sources. Extension of the SVQ to independent-component vector sources paves the way for the development of a scheme for the quantization of vector sources with *dependent* components, as we will show later.

Since the various components of the above mentioned vector source can have different distributions, an ESVQ for an m-dimensional vector source is derived from m different variable-length scalar quantizers $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m$ — each tailored to the distribution of the corresponding subsource. Let $\mathcal{Q}_i \equiv \{q_{1i}, q_{2i}, \ldots, q_{n_i i}\}$ denote the set of quantization levels of the i^{th} quantizer $\mathcal{S}_i$ for $i \in J_m$ and $\mathcal{L}_i \equiv \{\ell_{1i}, \ell_{2i}, \ldots, \ell_{n_i i}\}$ be the corresponding set of positive integer lengths assigned to the quantization levels. Also, let $\mathcal{Q} \equiv \mathcal{Q}_1 \cup \mathcal{Q}_2 \cup \cdots \cup \mathcal{Q}_m$

and $\mathcal{L} \equiv \mathcal{L}_1 \cup \mathcal{L}_2 \cup \cdots \cup \mathcal{L}_m$ denote the set of all quantization levels and the set of all lengths, respectively. The ESVQ is defined as follows.

Definition 2: An m-dimensional ESVQ $\mathcal{V}$ derived from scalar quantizers $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m$ is a vector quantizer with a codebook $\mathcal{Z}$ given as,

$$\mathcal{Z} = \{\mathbf{z} = (z_1, z_2, \ldots, z_m) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \cdots \times \mathcal{Q}_m : \\ \ell_{f_1(z_1)1} + \ell_{f_2(z_2)2} + \cdots + \ell_{f_m(z_m)m} \leq L\}, \tag{2}$$

where the threshold L governs the rate mr (in bits per m-dimensional vector) of the quantizer, and the index function $f_i : \mathcal{Q}_i \to J_{n_i}$ is defined by, $f_i(q_{ji}) = j,\ j \in J_{n_i},\ i \in J_m$.

Remarks

1. According to the above definition, the ESVQ is unchanged if: (i) an integer k_i is added to all the lengths in $\mathcal{L}_i,\ i \in J_m$, (assuming that the resulting lengths are still positive) and $k = \sum_{i=1}^{m} k_i$ is added to the threshold L; and (ii) all the lengths in $\mathcal{L}$ $(= \mathcal{L}_1 \cup \mathcal{L}_2 \cup \cdots \cup \mathcal{L}_m)$ are multiplied by a positive rational s (assuming that the resulting lengths are still integers) and the threshold L is also multiplied by s.
2. It is interesting to note that in the case of ESVQ, a permutation of any codevector in general does not result in a codevector. This is because the different components of the codevector belong to different alphabets. Hence the idea of structured vector quantization can be extended to sources for which permutation coding does not work well.
3. Weighted pyramid and elliptical vector quantizers [11] are extensions of the pyramid and spherical vector quantizers, respectively, to independent-component vector sources. The scalar subsources of the vector source in this case have different variances but same distributions (Laplacian for weighted pyramid and Gaussian for elliptical vector quantizers). The weighted pyramid vector quantizer is essentially a special case of the ESVQ when all the scalar quantizers are uniform and $\ell_{ji} = w_i|q_{ji}|,\ j \in J_{n_i}, i \in J_m$. Here w_i is the weight associated with the i^{th} subsource. The elliptical vector quantizer is similarly a special case of the ESVQ when all scalar quantizers are uniform and $\ell_{ji} = w_i q_{ji}^2$.

A procedure for the design of the ESVQ is presented in the next section. Some of the steps in this case are straightforward extensions of the SVQ [1] but are presented here for the sake of completeness.

IV. Design of the ESVQ

Designing a rate r bits/sample ESVQ for a given m-dimensional independent-component vector source amounts to determining the quantization levels $\mathcal{Q}$, the lengths $\mathcal{L}$ and the threshold L. As in the case of the SVQ [1], this can be done by a two-step iterative process. The first step (Step A) determines the quantization levels $\mathcal{Q}$ given $\mathcal{L}$ and L. The second step (Step B) determines the lengths $\mathcal{L}$ and threshold L given $\mathcal{Q}$. If both these steps are optimal with respect to some distortion measure, then each iteration reduces the average distortion and convergence to a locally optimal design is guaranteed. While Step A can be optimally performed for some important distortion metrics, the optimal solution of Step B is not known and a suboptimal solution is presented here. This does not guarantee the convergence of the design algorithm but results in good suboptimal designs. The performance of the ESVQ designed using this algorithm is sensitive to the initial choice of $\mathcal{Q}$, $\mathcal{L}$ and L in the iterative process. For the quantizers designed here, we started with uniformly spaced quantization levels for each $\mathcal{Q}_i$, $i \in J_m$. The spacing between the levels was chosen such that the sum of the output entropies of $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m$ (assuming quantization thresholds to be midway between the levels) equals the rate mr bits/vector. For this choice of $\mathcal{Q}$ the initial values of $\mathcal{L}$ and L were assigned in accordance with Step B.

1. Step A

In this step, we determine $\mathcal{Q}$ for a given $\mathcal{L}$ and L. The procedure used here is a simple extension of that for the SVQ [1] and is itself performed iteratively. The non-negative, single-letter distortion measure between the source output x and its reproduction y is denoted by $d(x, y)$. A long training set of N source vectors is generated and this set is used to characterize the m-dimensional probability density function (*pdf*) of the vector source. Let $\mathcal{X} = \{\mathbf{x}_j,\ j \in J_N\}$ denote this set where $\mathbf{x}_j \equiv (x_{1j}, x_{2j}, \ldots,\ x_{mj})$.

Algorithm

0. For given $\mathcal{L}$, L, the training sequence $\mathcal{X}$ and stopping threshold $\epsilon > 0$, select the initial set of quantization levels $\mathcal{Q}^0 \equiv \mathcal{Q}_1^0 \cup \mathcal{Q}_2^0 \cup \cdots \cup \mathcal{Q}_m^0$ as the optimal solution for $\mathcal{Q}$ in the previous visit to Step A, and set the iteration index $k = 0$. The procedure for choosing $\mathcal{Q}^0$ for the first visit to Step A has already been described above.
1. Quantize each vector in $\mathcal{X}$ using the ESVQ defined by $(\mathcal{Q}^k, \mathcal{L}, L)$. Let $\mathcal{Y}_k = \{\mathbf{y}_j^k,\ j \in J_N\}$ represent the resulting set of quantized vectors where $\mathbf{y}_j^k \equiv (y_{1j}^k, y_{2j}^k, \ldots,\ y_{mj}^k) \in$

$\mathcal{Q}_1^k \times \mathcal{Q}_2^k \times \cdots \times \mathcal{Q}_m^k$ for $j \in J_N$. The average per-vector distortion D^k associated with this quantization operation can be expressed as,

$$D^k = \frac{1}{N} \sum_{j=1}^{N} \sum_{i=1}^{m} d(x_{ij}, y_{ij}^k) = \frac{1}{N} \sum_{i=1}^{m} \sum_{l=1}^{n_i} \sum_{\substack{j: y_{ij}^k = q_{li}^k \\ j \in J_N}} d(x_{ij}, q_{li}^k). \tag{3}$$

2. Update the set of quantization levels to $\mathcal{Q}^{k+1}$ where

$$q_{li}^{k+1} = \operatorname*{argmin}_{q} \sum_{\substack{j: y_{ij}^k = q_{li}^k \\ j \in J_N}} d(x_{ij}, q), \; l \in J_{n_i}, \; i \in J_m. \tag{4.a}$$

It is straightforward to show that $\mathcal{Q}^{k+1}$ minimizes D^k and hence reduces distortion over $\mathcal{Q}^k$. In the important special case of squared-error distortion ($d(x,y) = (x-y)^2$), it is easily shown that (4.a) reduces to

$$q_{li}^{k+1} = \left[\sum_{\substack{j: y_{ij}^k = q_{li}^k \\ j \in J_N}} 1 \right]^{-1} \left[\sum_{\substack{j: y_{ij}^k = q_{li}^k \\ j \in J_N}} x_{ij} \right], \; l \in J_{n_i}, \; i \in J_m. \tag{4.b}$$

Thus, for the squared-error distortion measure, the above equation corresponds to taking all the input samples that quantize to q_{li}^k (using the ESVQ defined by $(\mathcal{Q}^k, \mathcal{L}, L)$) and updating this quantization level to q_{li}^{k+1} which is the mean of all those samples. Similarly, for the absolute-error distortion ($d(x,y) = |x-y|$), each quantization level q_{li}^k is updated to the median of all the input samples that are quantized to it.

3. Set $k = k+1$ and quantize each vector in $\mathcal{X}$ using the SVQ defined by $(\mathcal{Q}^k, \mathcal{L}, L)$ and compute the corresponding average distortion D^k. If $(D^{k-1} - D^k)/D^{k-1} \leq \epsilon$, stop with $\mathcal{Q} = \mathcal{Q}^k$; else go to Step 2.

Since the updating in Step 2 above can only reduce the average distortion, the above algorithm converges for a given $(\mathcal{L}, L)$ resulting in $\mathcal{Q}^{opt}(\mathcal{L}, L)$.

2. Step B

In this step, we determine $\mathcal{L}$ and L for a given $\mathcal{Q}$. As mentioned before, the optimal solution for this step is not known. In what follows we present a suboptimal solution for determining $\mathcal{L}$. After obtaining $\mathcal{L}$, the threshold L can be chosen such that the ESVQ has the desired output rate.

2.a. *Determination of $\mathcal{L}$*

For the SVQ design, two methods for determining $\mathcal{L}$ given $\mathcal{Q}$ are described in [1]. While the first method (Method 1) performs well for the SVQ, there is an important reason for not using a similar method for the ESVQ. The lengths in [1], for a given $\mathcal{Q}$, are assigned based on probabilities of quantization to the nearest level using the scalar quantizer $\mathcal{S}$. This corresponds to the quantization thresholds being placed midway between the quantization levels of $\mathcal{S}$. A similar procedure for ESVQ will result in a set of lengths $\mathcal{L}_i$ of the quantizer $\mathcal{S}_i$ that is determined only by the probability distribution of the i^{th} source component. Hence the lengths associated with one quantizer are not dependent upon those of any other quantizer. This implies that there is no way of placing greater importance on the quantization of some (possibly larger variance) components of the vector source over the others. As an example consider a two-dimensional source having zero mean Gaussian components with variances 1 and 100. Also let $\mathcal{Q}_1 = \{q_1, q_2, \ldots, q_n\}$ and $\mathcal{Q}_2 = \{10q_1, 10q_2, \ldots, 10q_n\}$. This will result in identical sets of lengths $\mathcal{L}_1$ and $\mathcal{L}_2$. Clearly, a more desirable length assignment is one in which most length values in $\mathcal{L}_1$ are relatively large. To overcome this problem we use the following algorithm that is a generalization of the asymptotically optimal (for large m) solution of $\mathcal{L}$ for the SVQ (Method 2 in [1]).

First we determine the set of quantization thresholds $\mathcal{T}_i \equiv \{t_{0i}, t_{1i}, \ldots, t_{n_i i}\}$ for each scalar quantizer $\mathcal{S}_i$ ($i \in J_m$) with quantization levels $\mathcal{Q}_i$, such that the expected distortion is minimized given that the sum of the output entropies of the quantizers is no greater than the rate mr bits/vector. As described later, the probabilities of the quantization regions defined by these thresholds are used to determine the lengths in $\mathcal{L}$. Hence, the problem here is to determine the set of quantization thresholds $\mathcal{T} \equiv \mathcal{T}_1 \cup \mathcal{T}_2 \cup \cdots \cup \mathcal{T}_m$ such that the distortion

$$D(\mathcal{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n_i} \int_{t_{(j-1)i}}^{t_{ji}} d(x, q_{ji}) p_i(x) dx, \tag{5.a}$$

is minimized subject to the constraint on the output entropy

$$H(\mathcal{T}) = -\sum_{i=1}^{m} \sum_{j=1}^{n_i} p_{ji} \log_2 p_{ji} \leq mr, \tag{5.b}$$

where $p_i(x)$ is the *pdf* of the i^{th} scalar subsource and $p_{ji} = \int_{t_{(j-1)i}}^{t_{ji}} p_i(x)dx$ is the probability of quantizing to the j^{th} level of $\mathcal{S}_i$ when scalar quantization of the i^{th} scalar subsource is performed. This is similar to the design of an ECSQ [3] and there are several ways to solve for $\mathcal{T}$ including the one given in Section III.B.2 of [1]. If the source *pdf*s are not known, a training sequence based approach can be adopted and the integrals in (5) can be replaced by the corresponding sums over the training sequence. As in the case of the SVQ, once the quantization thresholds are determined the lengths ℓ_{ji} are evaluated as,

$$\ell_{ji} = [b \log_2(1/p_{ji})], \; j \in J_{n_i}, \; i \in J_m, \tag{6}$$

where the square brackets denote rounding off to the nearest integer and b is a real number that determines the effective round-off error. Larger b implies a smaller effective error but a higher implementation complexity. The effect of b on the complexity and performance of the SVQ is discussed in [1] and similar arguments hold for the ESVQ.

Even though the ESVQ does not explicitly allocate bits among the various components of the vector source, the bit (entropy) allocation in the above algorithm is implicit in the way the lengths are assigned to the quantization levels of the different scalar quantizers.

2.b. Determination of the threshold L

Now that $\mathcal{L}$ has been determined, the threshold L is obtained by counting the grid-points (starting with the ones that have the smallest total length) until 2^{mr} points are counted; the largest total length for which all grid-points of that length are included in this collection will determine the value of L. An algorithm to accomplish this is described below.

Let N_i^j represent the number of distinct i-vectors $(v_1, v_2, \ldots, v_i) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \cdots \times \mathcal{Q}_i$ such that their total length $\ell_{f_1(v_1)1} + \ell_{f_2(v_2)2} + \cdots + \ell_{f_i(v_i)i} = j$. Then N_i^j satisfies the recursive equation $N_i^j = \sum_{l=1}^{n_i} N_{i-1}^{j-\ell_{li}}$, $\forall \; i \in J_m$ where $N_i^j = 0$ for $j < 0$ and $N_0^0 = 1$. The number C_m^j of grid-points with a total length less than or equal to j is now given by $C_m^j = \sum_{k=1}^{j} N_m^k$. The threshold L can therefore be evaluated as, $L = \max\{j : \; C_m^j \leq 2^{mr}\}$. This choice of L guarantees that there will be at most 2^{mr} vectors in the codebook and hence each codevector can be encoded by an mr-bit binary codeword.

3. ESVQ design algorithm

The ESVQ design consists of successive updating of $\mathcal{Q}$ and $(\mathcal{L}, L)$ by applying Steps A and B, respectively. As noted before, the convergence of this algorithm is not guaranteed because of the suboptimality of Step B. For cases where the solution does not converge, a fixed number of iterations are performed from which the best quantizer is chosen. As with the SVQ, for those cases where the solution converged, the convergence does not imply any optimality or local optimality and can be attributed to the discrete (integer) nature of the lengths in $\mathcal{L}$. The design algorithm for the ESVQ is presented below.

Algorithm

0. Let $\epsilon > 0$, $k_{\max}$ and $k_{\min}$ denote the stopping threshold, the maximum and the minimum number of iterations. For a given training sequence $\mathcal{X}$ design an initial ESVQ consisting of $\mathcal{Q}^0$, $\mathcal{L}^0$ and L^0 as explained in the beginning of this section; set the iteration index $k = 0$; set initial average distortion $D^0 = \infty$.
1. For fixed $\mathcal{L}^k$ and L^k, use Step A (with $\mathcal{Q}^k$ as initial set of quantization levels) to obtain $\mathcal{Q}^{k+1} \equiv \mathcal{Q}^{opt}(\mathcal{L}^k, L^k)$. Compute the average distortion D^{k+1} associated with the triple $(\mathcal{Q}^{k+1}, \mathcal{L}^k, L^k)$. If $(D^k - D^{k+1})/D^k < \epsilon$ and $k \geq k_{\min}$, stop with $(\mathcal{Q}^{k+1}, \mathcal{L}^k, L^k)$ defining the ESVQ, else if $k = k_{\max}$, stop and select the ESVQ as that which resulted in the smallest average distortion among all iterations up to $k_{\max}$; else, set $k = k + 1$.
2. For $\mathcal{Q}^k$, determine from Step B, the set of lengths $\mathcal{L}^k$ and the threshold L^k. Go to Step 1.

V. Implementation of the ESVQ

The implementation of the ESVQ defined by $\mathcal{Q}$, $\mathcal{L}$ and L essentially involves two steps: (i) finding the vector in the codebook closest to the input vector (codebook search) and (ii) assignment of a binary codeword to the selected codevector at the transmitter and finding the codevector corresponding to the received codeword at the receiver (codevector encoding/decoding). Given the special structure of the ESVQ codebook, both these steps can be efficiently performed as described below.

1. Codebook search

Let D_i^j be the minimum squared-error distortion that results when the first i components of the input m-vector $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ are quantized to any of the i-vectors $\mathbf{z}_i = (z_1, z_2, \ldots, z_i) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \cdots \times \mathcal{Q}_i$ such that the total length of $\mathbf{z}_i$ is j, that is, $\ell_{f_1(z_1)1} + \ell_{f_2(z_2)2} + \cdots + \ell_{f_i(z_i)i} = j$. Also, let $\mathbf{z}_i^j$ represent the i-vector that results in the minimum distortion D_i^j. The distortion D_{i+1}^j is then recursively given as,

$$D_{i+1}^j = \min_{k \in J_{n_{i+1}}} \left[D_i^{j-\ell_{k(i+1)}} + (x_{i+1} - q_{k(i+1)})^2 \right], \; i \in \{0\} \cup J_{m-1}, \tag{7}$$

where $D_i^j = \infty$ for $j \leq 0$ or $i \leq 0$ but $D_0^0 = 0$.

If the above is minimized for $k = k'$, then the corresponding minimum distortion vector $\mathbf{z}_{i+1}^j$ is given by the recursive equation $\mathbf{z}_{i+1}^j = (\mathbf{z}_i^{j-\ell_{k'(i+1)}}, q_{k'(i+1)})$. We can solve (7) using a dynamic programming algorithm [13] similar to the Viterbi algorithm [14] and determine D_m^j, $\forall\, j \in J_L$. The minimum distortion $D_{\min}$ is then given as, $D_{\min} = \min_{j \in J_L} D_m^j$. If $j = j'$ minimizes this distortion, then $\mathbf{z}_m^{j'}$ gives the desired codevector.

Although the order in which indices are assigned to the scalar subsources to form a vector source does not affect the performance of the ESVQ, it does make a difference on its implementation complexity. From an implementation standpoint, it is advantageous to label the subsources so that their variances are in a decreasing order, i.e., the first component of the vector has the largest variance, the second component has the next largest variance and so on. This is because a large variance subsource (i^{th}) typically has a large number n_i, of quantization levels associated with it. If such a subsource is assigned a smaller index, (7) for this source will have to be computed for a smaller range of values of j, resulting in reduced implementation complexity.

The codebook search algorithm described above has a considerably higher complexity than simple component-by-component scalar quantization of the source m-vector using the scalar quantizer bank $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_m$. The overall search complexity can hence be reduced by first performing this simple component-by-component quantization and checking the total length of the resulting vector (grid-point). If the total length is no greater than the threshold L, this is the required codevector; if not, the codebook search algorithm can be used.

2. Encoding of codevectors

There are 2^{mr} codevectors in the codebook $\mathcal{Z}$ of the ESVQ and the encoder is a mapping that assigns mr-bit binary numbers to the codevectors in a one-to-one manner. The following algorithm implements one such mapping.

To every codevector $\mathbf{z} = (z_1, z_2, \ldots, z_m) \in \mathcal{Z}$ is assigned a unique m-tuple $\mathcal{N}(\mathbf{z}) \equiv (f_m(z_m), f_{m-1}(z_{m-1}), \ldots, f_1(z_1))$. All the possible m-tuples can now be ordered in a dictionary order. That is, one m-tuple is smaller than the other if either its first component is smaller than the first component of the other, or if their first components are equal, the second component is smaller than the second component of the other and so on. The encoder function $E : \mathcal{Z} \rightarrow \{0\} \cup J_{2^{mr}-1}$ is defined as, $E(\mathbf{z}) = \sum_{\mathbf{w} \in \mathcal{Z}} X_{\mathbf{zw}}$, where $X_{\mathbf{zw}} = 1$ if $\mathcal{N}(\mathbf{w}) < \mathcal{N}(\mathbf{z})$ and 0 otherwise. In other words, $E(\mathbf{z})$ is the number of codevectors in $\mathcal{Z}$ that are "smaller" than $\mathbf{z}$ (smaller in the sense $\mathcal{N}(\mathbf{w}) < \mathcal{N}(\mathbf{z})$). We can also write $E(\mathbf{z}) = \sum_{k=1}^{m} E_k$, where E_k is the number of codevectors $\mathbf{w}$ such that the m-tuples $\mathcal{N}(\mathbf{w})$ and $\mathcal{N}(\mathbf{z})$ are equal in their first $(k-1)$ components while the k^{th} component of $\mathcal{N}(\mathbf{w})$ is smaller than that of $\mathcal{N}(\mathbf{z})$. The mr-bit binary codeword associated with the codevector $\mathbf{z}$ is then the binary representation of $E(\mathbf{z})$.

Let C_i^j represent the number of distinct i-vectors $(z_1, z_2, \ldots, z_i) \in \mathcal{Q}_1 \times \mathcal{Q}_2 \times \cdots \times \mathcal{Q}_i$ with total length no greater than j. Also, define $C_0^j = 0$ if $j < 0$ and 1 otherwise. It is now easy to see that $E_k = \sum_{j=1}^{f_k(z_k)-1} C_{m-k}^{L-L_{k-1}-\ell_{jk}}$ for $k \in J_m$, where $L_0 = 0$ and $L_i = \sum_{j=m-i+1}^{m} \ell_{f_j(z_j)j}$, $i \in J_m$. Now we can write $E(\mathbf{z})$ as,

$$E(\mathbf{z}) = \sum_{k=1}^{m} \sum_{j=1}^{f_k(z_k)-1} C_{m-k}^{L-L_{k-1}-\ell_{jk}} . \tag{8}$$

Note that C_i^j can be evaluated for $i \in J_m$ and $j > 0$ by expressing it in terms of N_i^j (Section IV) as, $C_i^j = \sum_{k=1}^{j} N_i^k$. For a fast implementation of this algorithm the C_i^j can be evaluated once and stored in memory.

The decoder mapping is the inverse of the encoder mapping. The decoding algorithm is not described here but can easily be determined from the encoding algorithm.

The complexity of implementation of the ESVQ is similar to that of the SVQ, and as shown in [1], both the storage and the worst-case computational requirements are exponential in r but only polynomial in m. This makes the structured vector quantizers much easier to implement than LBG vector quantizers [7] where the complexity is exponential in mr.

In the next section, it is shown that the ESVQ can be used in conjunction with certain block transformations to quantize sources with memory.

VI. Quantization of Sources with Memory

Quantization of correlated sources using block transforms has been studied in the past and is a popular technique for image and speech coding [8],[9],[15]. The transformations used for this purpose (such as the Karhunen-Loève transform and the discrete cosine transform) have correlation removing property and the output vectors of these transformations approximately fit the model of the independent-component vector source described here. The ESVQ is hence a good candidate for the efficient quantization of the transformation output vector.

A linear transformation represented by an $m \times m$ matrix $\mathbf{A}$ can be used to transform a block of m samples (or an m-vector) $\mathbf{x}^T \equiv (x_1, x_2, \ldots, x_m)$ from a stationary correlated source. The transformation applied to $\mathbf{x}$ results in the output vector $\mathbf{c} = \mathbf{A}\mathbf{x}$. If $\mathbf{B}$ is the inverse transformation matrix, then $\mathbf{x} = \mathbf{B}\mathbf{c}$. The transformation $\mathbf{A}$ is called orthonormal if $\mathbf{A}^{-1} = \mathbf{A}^T$. An orthonormal transformation $\mathbf{A}$ corresponds to a rotation of the coordinate system such that: (i) the Euclidean distance between points (vectors) is preserved, i.e., for two vectors $\mathbf{x}_1$, $\mathbf{x}_2$ and the corresponding transformation output vectors $\mathbf{c}_1$, $\mathbf{c}_2$, respectively, $(\mathbf{c}_1 - \mathbf{c}_2)^T(\mathbf{c}_1 - \mathbf{c}_2) = (\mathbf{x}_1 - \mathbf{x}_2)^T(\mathbf{x}_1 - \mathbf{x}_2)$, and (ii) the sum of the variances is preserved, i.e., $\sum_{i=1}^{m} \sigma_{c_i}^2 = \sum_{i=1}^{m} \sigma_{x_i}^2$, where $\sigma_{c_i}^2$ and $\sigma_{x_i}^2$ are the variances of the i^{th} transform coefficient and the input sample, respectively.

In block transform coding, the transformation output vector $\mathbf{c}$ is quantized to obtain $\mathbf{c}'$. At the receiver, the inverse transformation $\mathbf{B}$ is performed on $\mathbf{c}'$ to obtain the reconstructed version $\mathbf{x}'$ of $\mathbf{x}$. The transformation chosen for this purpose is usually orthonormal and represents the source vector in a special coordinate system — one in which the resulting components are uncorrelated and can therefore be efficiently quantized using scalar quantizers. Thus, the decorrelating transformation reduces the redundancy present in the source vector and offers the possibility to use scalar quantizers making the scheme simple to implement. Because of the Euclidean distance preserving nature of these transformations, for the squared-error distortion measure, the reconstruction error of each source vector is the same as the quantization error of the transformed vector and therefore the quantizers can be designed to minimize the expected value of the latter.

It is shown in [15] that for Gaussian input (in the limit of fine quantization) the transform coding gain[2] ${}^{m}G_{TC}$ realized by an m^{th} order orthonormal transformation over direct scalar quantization of the source output is given as,

$$ {}^{m}G_{TC} = \frac{\frac{1}{N}\sum_{i=1}^{m} \sigma_{c_i}^2}{\left[\prod_{i=1}^{m} \sigma_{c_i}^2\right]^{\frac{1}{N}}} \,. \tag{9} $$

The coding gain therefore is the ratio of the arithmetic mean to the geometric mean of the transform coefficient variances.

The optimal m^{th} order transformation for stationary Gaussian sources is the Karhunen-Loève transform (KLT) which maximizes the coding gain ${}^{m}G_{TC}$ over all possible linear transformations [8]. The KLT results in a diagonal autocovariance matrix $\mathbf{R_{cc}} \equiv E[\mathbf{CC}^T] = \mathbf{AR_{xx}A}^T$, where $\mathbf{R_{xx}}$ is the m^{th} order symmetric positive-definite source autocovariance matrix. Let λ_i, $i = 1, 2, \ldots, m$, be the m distinct eigenvalues of $\mathbf{R_{xx}}$ and $\mathbf{I}_i$, $i = 1, 2, \ldots, m$, be the corresponding eigenvectors scaled such that $\mathbf{I}_i^T\mathbf{I}_j = \delta_{ij}$. The KLT matrix has as its i^{th} row the eigenvector $\mathbf{I}_i$, $i \in J_m$, of $\mathbf{R_{xx}}$ and the resulting diagonal matrix $\mathbf{R_{cc}}$ has as its diagonal elements the eigenvalues λ_i, $i = 1, 2, \ldots, m$, of $\mathbf{R_{xx}}$. The coding gain ${}^{m}G_{KLT}$ of the transformation is given as,

$$ {}^{m}G_{KLT} = \frac{\frac{1}{N}\sum_{i=1}^{m} \lambda_i}{\left[\prod_{i=1}^{m} \lambda_i\right]^{\frac{1}{N}}} \,. \tag{10} $$

For the KLT, the diagonal nature of the $\mathbf{R_{cc}}$ matrix implies that the transformation output vector has uncorrelated components. This is intuitively satisfying because it says that the optimal transformation is the one that decorrelates the source vector (removing redundancy) before quantizing. For Gaussian samples, uncorrelatedness also implies independence and hence the output vector $\mathbf{c}$ has independent components that can be efficiently quantized using the ESVQ.

Although the KLT is the optimal transformation in terms of coding gain, it is an input-dependent transformation and requires the knowledge of the source autocovariance matrix

[2] The gain here refers to the factor by which the quantization error variance is reduced. The corresponding gain in the signal-to-noise ratio is given as $10\log_{10}({}^{m}G_{TC})$ in dB.

which may not be available. For such cases one can use suboptimal source-independent transformations provided the source is stationary. One such transformation is the discrete cosine transform (DCT) which for sources such as speech and imagery has coding gains very close to that of the KLT [15].

Several researchers have studied block transform coding for the quantization of stationary sources with known correlation properties. Huang and Schultheiss in [8] have applied the KLT to correlated Gaussian sources. The transformation output vector is then quantized using a bank of fixed-length scalar LMQs. The rate of each quantizer in the bank is determined optimally by a bit allocation algorithm. The vector components with larger variances receive more bits while some coefficients with small variances may not receive any bits. Farvardin and Lin have studied entropy-constrained quantization of the KLT coefficients for quantizing correlated Gaussian sources [9]. Since optimal entropy-constrained quantizers are hard to design, they have used entropy-coded uniform-threshold quantizers which are known to perform close to the optimum [3]. An optimal entropy allocation algorithm is used to allocate entropy among the various components of the vector. This results in improved performance over the scheme of Huang and Schultheiss but at the cost of a variable-rate output.

We have studied the fixed-rate quantization of correlated Gaussian sources by using the ESVQ to quantize the KLT transformed vectors. This scheme is hereafter referred to as the KLT-ESVQ scheme. If the source being considered is Gaussian, the equiprobability density surfaces in the m-dimensional space are (concentric) generalized ellipses [11]. The KLT then corresponds to a rotation of the coordinate system so as to align it to the axes of these m-dimensional ellipses. Consider the following ESVQ for quantizing the KLT output vectors for a Gaussian source: (i) the scalar quantizers $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_m$ are uniform quantizers with equal step sizes, and (ii) the lengths ℓ_{ji} (for $j = 1, 2, \ldots, n_i$) associated with the i^{th} quantizer $\mathcal{S}_i$, $i \in J_m$, are given by, $\ell_{ji} = w_i q_{ji}^2$, where w_i is the reciprocal of the variance of the i^{th} component of the KLT output vector. This ESVQ is similar in spirit to the elliptical vector quantizer in [11] and for an appropriate value of the threshold L, places the codevectors (on a cubic lattice) inside and on the surface of the nearly unit probability (for large m) m-dimensional generalized ellipse. This is however not necessarily the best ESVQ that can be designed for quantizing such sources, especially when m is not large. An improvement in performance could result if this quantizer is used as the initial choice in the design algorithm of Section IV.

The results of the KLT-ESVQ scheme on first-order Gauss-Markov sources with correlation coefficients 0.8 and 0.9 are presented in Tables 1 and 2, respectively. The corresponding ESVQs were designed using the algorithm of Section IV with lengths effectively quantized to multiples of a quarter of a bit (corresponding to $b = 4$). A total of 50000 source samples were used in the simulations. The performance results for the Huang and Schultheiss (HS) scheme, Farvardin and Lin (FL) scheme, optimum entropy coded predictive quantizer (ECPQ) [16] (where available) and LBGVQ [7],[17] are also included in the tables for comparison. Due to the design complexity, the LBGVQ results reported here are for a codebook of size 1024 vectors except at the rate of 3 bits/sample in which case the codebook size is 512 vectors.

These tables show that as expected, the performance of the KLT-ESVQ scheme improves with block-length. This is due to two reasons: (i) the performance of the ESVQ itself improves with block-length because the SVQ idea is based on the law of large numbers, and (ii) the coding gain of the KLT improves with block-length [15]. The latter is also responsible for the better performance of the HS and FL schemes at larger block-lengths. As expected, the KLT-ESVQ scheme for a given encoding rate and block-length, performs better than the fixed-rate HS scheme and close to the variable-rate FL scheme. For a fixed block-length, the gap between the KLT-ESVQ and the FL scheme is larger at higher rates. This is consistent with the observation in [1] that at higher rates the SVQ requires a larger block-length to perform close to the optimal ECSQ.

From Table 1 it can be seen that at high rates (fine quantization), the performance of the variable-rate ECPQ is slightly better than the KLT-ESVQ scheme, while at low rates the latter does considerably better. This is because the ECPQ is known not to perform well at low rates but at high rates it performs close to (only 1.53 dB below) the rate-distortion limit [16]. Though for the same block-length and rate, vector quantizers perform better than the KLT-ESVQ scheme, the block-length of the KLT-ESVQ can be increased to get performance better than LBGVQ at the same rate. For example in Table 2, the KLT-ESVQ scheme at 1 bit/sample and block-length 32 performs 0.53 dB better than the 10-dimensional LBGVQ at the same rate. Note that the design and/or implementation of a rate 1 bit/sample LBGVQ with block-length 32 (having 2^{32} codevectors) is practically impossible. In addition, for the KLT-ESVQ it is possible to operate at higher bit-rates with a manageable complexity. We have reported in Tables 1 and 2 results up to 3 bits/sample with a block-length as large as 32, implying an effective codebook size of 2^{96}; for such a high bit-rate, the VQ block-length cannot exceed 5.

For Gauss-Markov sources, the gap between the performances of the HS and FL schemes is not much, leaving only a limited room for improvement. The ESVQ-based block transform coding scheme on other sources for which this gap is larger should result in higher gains over the HS scheme. It is well known [15],[18] that the two-dimensional discrete cosine transformation of image data results in close-to-Laplacian distribution of the transform coefficients (except for the "dc" coefficient). Since for Laplacian sources there is a significant gap between the performances of fixed- and variable-length scalar quantizers, the ESVQ offers a potential for significant improvement over fixed-length quantization of the transform coefficients.

VII. ESVQ versus SVQ

So far we have considered the quantization of independent-component vector sources with the ESVQ. However, these sources can also be quantized using a bank of SVQs (for scalar memoryless sources) as follows. Since the i^{th} component of the source m-vector can be assumed to come from a memoryless scalar subsource $\mathcal{X}_i$, $i \in J_m$, an appropriately designed SVQ can be used to efficiently quantize a block of m'_i output samples from this source. Given the block-size of the SVQ for each subsource, its rate-distortion performance can be determined and used for optimal rate allocation among the m subsources of the vector source. We now compare the ESVQ with this SVQ-based scheme, especially in the context of block transform quantization.

One obvious drawback of the SVQ-based system is the delay it introduces. If $m'_{\max}$ is the largest of the block-lengths of the m SVQs, then there is at least an $m'_{\max}$ vector coding delay introduced into the system. In contrast, the ESVQ introduces no such delay. Further, if the source vector is obtained by using a decorrelating m-dimensional transformation on a source with memory, the coding delay is $m \times m'_{\max}$ samples. While this delay may not be of any consequence in still image coding (where the vectors do not form a sequence in time), it can be important in some other applications and will limit the SVQ block-length and hence the performance.

In the absence of any constraints on delay or complexity, the asymptotic performances (for large block-lengths) of the ESVQ- and SVQ-based block transform quantization schemes are expected to be the same. However, as explained below, for practical systems that place a limit on the complexity, the ESVQ-based system has an edge. In many practical schemes that use the KLT or DCT, the low frequency components (subsources) of the transform vectors have variances significantly larger than those of the high frequency components. Thus, some subsources must be encoded with significantly higher rates than the average rate. Since the complexity of implementation of the SVQ is exponential in rate, the block-length of SVQs designed for these sources is usually small and limited by the allowed complexity. This is particularly undesirable because at a high rate the SVQ requires a larger block-length to perform close to the optimal variable-length encoded scalar quantizer [1]. For the ESVQ on the other hand, the larger variance subsources are quantized together with the smaller variance sources allowing larger block-lengths (than the SVQ) for a similar complexity. This means that for the ESVQ, even the large variance subsources can enjoy the benefits of a large block-length. For a given limit on complexity, in our experience, the ESVQ-based scheme usually outperforms the SVQ-based block transform coding scheme.

Table 3 compares the performance results (SNR in dB) of the 32-dimensional KLT-ESVQ scheme to the corresponding results of the SVQ-based system for a Gauss-Markov source with coefficient of correlation 0.9 . For these results, the maximum SVQ dimension was taken to be $m'_{\max} = 32$. Also, to place a limit on the complexity, the total rate of the SVQ was limited to 96 bits/block. Under the above constraints, SVQs with maximum possible block-lengths were designed. This table confirms the claim that for comparable complexities, the ESVQ-based block transform quantizer performs better than the SVQ-based scheme; the ESVQ also introduces a significantly smaller coding delay.

VIII. Summary and Conclusions

This paper extends the fixed-rate structured vector quantizer to vector sources for which each component is a stationary memoryless scalar subsource independent of the other components. A suboptimal design algorithm for the ESVQ was presented along with fast algorithms for codebook search and codevector encoding. It is shown that the ESVQ can be applied to the quantization of stationary sources with memory. This is done by first using the KLT or DCT to decorrelate a block of source data. The transform output vector is then quantized using the ESVQ. Numerical results were presented for first-order

Gauss-Markov sources and it was shown that the KLT-ESVQ scheme improves upon the performance of the fixed-rate HS scheme and performs close to the variable-rate FL scheme. It was also shown that the KLT-ESVQ scheme performs better than implementable vector quantizers (especially at high rates). Because of the large gap in the performances of fixed- and variable-length scalar quantizers for Laplacian sources, the suboptimal scheme that uses the ESVQ to quantize the DCT output vectors can potentially be used for the efficient quantization of image data [18].

References

1. R. Laroia and N. Farvardin, "A Structured Fixed-Rate Vector Quantizer Derived from a Variable-Length Encoded Scalar Quantizer," submitted to *IEEE Trans. Inform. Theory*, Aug. 1991.
2. H. Gish and J. N. Pierce, "Asymptotically Efficient Quantizing," *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 676-683, Sept. 1968.
3. N. Farvardin and J. W. Modestino, "Optimum Quantizer Performance for a Class of Non-Gaussian Memoryless Sources," *IEEE Trans. Inform. Theory*, Vol. IT-30, pp. 485-497, May 1984.
4. T. Berger, "Optimum Quantizers and Permutation Codes," *IEEE Trans. Inform. Theory*, Vol. IT-18, pp. 759-765, Nov. 1972.
5. S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Trans. Inform. Theory*, Vol. IT-28, pp. 129-137, March 1982.
6. J. Max, "Quantizing for Minimum Distortion," *IRE Trans. Inform. Theory*, Vol. IT-6, pp. 7-12, March 1960.
7. Y. Linde, A. Buzo and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. Commun.*, Vol. COM-28, pp. 84-95, Jan. 1980.
8. J. J. Y. Huang and P. M. Schultheiss, "Block Quantization of Correlated Gaussian Random Variables," *IEEE Trans. Comm. Syst.*, Vol. CS-11, pp. 289-296, September 1963.
9. N. Farvardin and F. Y. Lin, "Performance of Entropy-Constrained Block Transform Quantizers," *IEEE Trans. Inform. Theory*, Vol. 37, No. 5, pp. 1433-1439, Sept. 1991.
10. T. R. Fischer, "A Pyramid Vector Quantizer," *IEEE Trans. Inform. Theory*, Vol. IT-32, pp. 568-583, July 1986.

11. T. R. Fischer, "Geometric Source Coding and Vector Quantization," *IEEE Trans. Inform. Theory*, Vol. IT-35, pp. 137-145, Jan. 1989.
12. Rajiv Laroia, *Structured Fixed-Rate Vector Quantizers Derived from Variable-Length Encoded Scalar Quantizers*, Ph.D. Dissertation, Electrical Engineering Department, Univ. of Maryland, College Park, in preparation.
13. R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.
14. A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*, McGraw-Hill, 1979.
15. N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Application to Speech and Video*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1984.
16. N. Farvardin and J. W. Modestino, "Rate-Distortion Performance of DPCM Schemes for Autoregressive Sources," *IEEE Trans. Inform. Theory*, Vol. IT-31, pp. 402-418, May 1985.
17. R. Gray and Y. Linde, "Vector Quantizers and Predictive Quantizers for Gauss-Markov Sources," *IEEE Trans. Commun.*, Vol. COM-30, pp. 381-389, Feb. 1982.
18. R. C. Reininger and J. D. Gibson, "Distribution of the Two-Dimensional DCT Coefficients for Images," *IEEE Trans. Commun.*, Vol. COM-31, pp. 835-839, June 1983.

Rate	Block-Length						ECPQ	LBGVQ	D(R)
	1	4	8	16	24	32			
	KLT-ESVQ								
1	4.40	8.17	8.66	9.13	9.36	9.41	7.66	9.20	10.46
2	9.30	13.15	13.60	14.48	14.64	14.73	14.44	14.01	16.48
3	14.62	18.47	19.40	20.00	20.27	20.40	20.96	18.70	22.50
	HS								
1	4.40	7.90	8.39	8.69	8.81	8.85			
2	9.30	12.64	13.20	13.48	13.57	13.62			
3	14.62	17.78	18.44	18.69	18.77	18.81			
	FL								
1	4.58	8.47	9.02	9.30	9.39	9.44			
2	10.55	13.86	14.41	14.68	14.75	14.83			
3	16.56	19.84	20.42	20.69	20.78	20.84			

Table 1: Performance results (SNR in dB) for the quantization of a first-order Gauss-Markov source with coefficient of correlation 0.8 . The rate is in bits/sample. Here HS refers to the Huang-Schultheiss scheme [8] and FL refers to the Farvardin-Lin scheme [9]. The LBGVQ codebook size is 1024 vectors for $r = 1,\ 2$ bits/sample and 512 vectors for $r = 3$ bits/sample. The performance of the entropy-coded predictive quantizer (ECPQ) and the source distortion-rate function are also given.

Rate	Block-Length						ECPQ	LBGVQ	D(R)
	1	4	8	16	24	32			
	KLT-ESVQ								
1	4.40	10.04	11.21	11.84	12.05	12.17	10.01	11.64	13.23
2	9.30	15.16	16.54	17.23	17.42	17.50	NA	16.27	19.25
3	14.62	20.70	21.77	22.61	22.93	23.11	NA	20.59	25.27
	HS								
1	4.40	10.02	10.89	11.33	11.52	11.60			
2	9.30	14.90	15.69	16.14	16.29	16.37			
3	14.62	20.14	20.87	21.35	21.47	21.56			
	FL								
1	4.58	10.60	11.53	11.98	12.13	12.20			
2	10.55	15.98	16.85	17.30	17.46	17.53			
3	16.56	21.94	22.84	23.30	23.45	23.52			

Table 2: Performance results (SNR in dB) for the quantization of a first-order Gauss-Markov source with coefficient of correlation 0.9 . The rate is in bits/sample. Here HS refers to the Huang-Schultheiss scheme [8] and FL refers to the Farvardin-Lin scheme [9]. The LBGVQ codebook size is 1024 vectors for $r = 1,\ 2$ bits/sample and 512 vectors for $r = 3$ bits/sample. The performance of the entropy-coded predictive quantizer (ECPQ) and the source distortion-rate function are also given.

Rate	ESVQ	SVQ
1	12.17	12.01
2	17.51	17.18
3	23.11	22.79

Table 3: Comparison of the ESVQ and SVQ-based block transform (KLT) quantization of a first-order Gauss-Markov source with correlation coefficient 0.9 . The numbers indicate SNR in dB and the rate is given in bits/sample. The KLT block-size is 32 and the ESVQ block-length is also 32 ($m = 32$). The SVQ-based scheme uses 32 different SVQs as described in Section VII.